Tampereen ammattikorkeakoulu

# Prometheus virtuaalikoneympäristössä

## Valvontaa joustavasti

Jenni Tammenmaa

OPINNÄYTETYÖ
Toukokuu 2020

Tieto- ja viestintätekniikka
Ohjelmistotekniikka

**TIIVISTELMÄ**

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikka
Ohjelmistotekniikka

TAMMENMAA, JENNI:
Prometheus virtuaalikoneympäristössä
Valvontaa joustavasti

Opinnäytetyö 27 sivua
Toukokuu 2020

---

Tässä opinnäytetyössä tarkastellaan virtuaalikoneympäristöä ja sen valvontatarpeita. Tarpeisiin hyödynnettiin Prometheus-valvontaohjelmaa ja sen lisäosia. Tämän lisäksi visualisointi ratkaistiin Grafana-visualisointiohjelmalla.

KVM-ympäristössä toimiville Linux-koneille täytyi löytää valvontaratkaisu. Prometheus-valvontaohjelma on avoimen lähdekoodin valvontajärjestelmä, joka toimii tietoa lukevilla ja kuljettavilla lisäosilla. Tämä ohjelma valittiin työssä käytettäväksi, sillä toimeksiantajalla oli tarve tarkastella sitä asiantuntijatasolla. Prometheus asennettiin valvomaan kolmea virtuaalikonetta ja niiden suoritusta.

Suorituksen valvomisen lisäksi tarkastellaan Prometheuksen muita ominaisuuksia. Lisäosia tulee asentaa aina sitä mukaa, kun ominaisuuksia lisätään. Jotta kaikkien osien käyttö olisi turvallista, täytyisi suojauksen olla kunnossa itse valvontaohjelmassa. Myös ulkopuolisiin koneisiin asennetut lisäosat täytyisi turvallisuuden puolesta tarkistaa, jotta valvontaohjelmaan menevä data ei vahingoittaisi sen toimintaa.

Visualisointi ratkaistiin Grafanalla, jossa on mahdollisuus luoda datalähteistä erilaisia kojelautoja. Prometheuksen keräämän datan visualisointiin löytyi useampi erilainen kojelauta valmiina ja näistä löydettiin tähän tarpeeseen sopiva. Se oli suunniteltu näyttämään koneiden suoritusta. Visualisointi näytti oikeanaikaista dataa siitä, mitä koneissa sillä hetkellä tapahtui. Visualisointi pystyttiin rajaamaan myös yhteen koneeseen.

Jatkokehityksessä tulee ottaa huomioon suojaus, hälytykset sekä tietoliikenne. Jatkossa ohjelman käyttöä tullaan tarkastelemaan erilaisissa tilanteissa ja valvontatarpeissa. Prometheus ei ole osoittautunut paremmaksi tai huonommaksi kuin muut valvontajärjestelmät. Se kuitenkin eroaa muista ja onkin joihinkin tarpeisiin tehokas.

---

Asiasanat: prometheus, valvonta, linux, virtualisointi

## ABSTRACT

Tampere University of Applied Sciences
Degree Programme in ICT Engineering
Software Engineering

TAMMENMAA, JENNI:
Prometheus in a Virtual Machine Environment
Flexible Monitoring

Bachelor's thesis 27 pages
May 2020

---

The purpose of this thesis was to examine a virtual machine environment and the requirements for monitoring. Prometheus monitoring system was used to meet these needs, while visualization was implemented with Grafana visualization software.

Virtual machines that were hosted in a KVM-environment had to be found a monitoring solution. Prometheus is an open-source monitoring software, which operates on extensions which read and transport data. This software was used because it was unknown and to achieve expertise. Prometheus was installed to monitor three virtual machines and their performance.

In addition to the performance of the virtual machines other features had to be studied in Prometheus. Extensions should be installed as new features are added. So that the safety of all the parts that are used in the set is ensured, should the protection be in order in the monitoring software itself. Also, the extensions installed on machines outside of the Prometheus server, should be checked for safety, so that the data transferred does not harm the system.

Visualization issue was solved with Grafana, which has an option to create dashboards out of data sources. The data collected by Prometheus had several dashboards designed to display the data and the one used, was designed to show the performance of a machine hardware. The visualization showed real-time data about what was going on in the virtual machines. The dashboard could be limited to show data from just one source.

Further development should take protection, alerts, and communication into account. In the future the usage of the software will be observed in different situations and monitoring needs. Prometheus has not shown to be any better or worse than other monitoring solutions. It does, however, differ from others and is the most effective choice in some cases.

---

Key words: prometheus, monitoring, linux, virtualization

**SISÄLLYS**

## LYHENTEET JA TERMIT

| | |
|---|---|
| KVM | Kernel-based Virtual Machine, ytimeen perustuva tuki virtuaalikoneympäristölle |
| Linux | Linux-ydintä käyttävä käyttöjärjestelmä |
| HP | Hewlett-Packard Company, jonka yksi pääaloista on myydä tietotekniikan tuotteita |
| Hypervisor | Ohjelma, laitteisto tai näiden yhdistelmä, joka toimii alustana useammalle virtuaalikoneelle |

# 1 JOHDANTO

Tässä opinnäytetyössä tarkastellaan virtuaalikoneympäristöä ja sen valvontatarpeita, sekä luodaan esimerkkitilanne, jossa käytetään Prometheus-valvontaohjelmaa. Toimeksiantajana toimi Netum Oy, ja tavoitteena oli saavuttaa kyseisen valvontaohjelman asiantuntijuus.

Tämän työn tarkoituksena oli tutkia Prometheus-valvontaohjelman toimivuutta virtuaalikoneympäristössä ja tarkastella sen mahdollisuuksia erilaisissa valvontatapauksissa. Työssä tehtiin KVM-ympäristöön valvontaesimerkki, mitä voidaan hyödyntää tulevaisuuden valvontatarpeissa. KVM-ympäristö on rakennettu aikaisemmin, ja palveli tässä tapauksessa hyvin.

Työn kulku alkoi muutaman virtuaalikoneen asennuksesta. Työ jatkui aina valvontaohjelman asennuksesta valvottavien kohteiden konfigurointiin. Visualisointi hoidettiin lopuksi visualisointiohjelmalla. Prometheuksen lisäosia tutkittiin ja niiden käyttöä tarkasteltiin, mutta mitään niistä ei käytetty lopulliseen tulokseen. Koneiden valvonta onnistui moitteetta katsomalla ainoastaan graafeja.

Opinnäytetyön aihe syntyi työskentelyyn tarvittavan ympäristön valvomattomuudesta. Valvonnan tärkeys nykyään korostuu päivittäisessä toimimisessa jokaisella ja vaihtoehtoja valvonnan toteuttamiseen löytyy hyvinkin paljon. Tässä työssä kokeiltiin jotain, mihin ei suoraan löydy etsimällä vastausta eli miten vähemmän suosittu valvontaohjelma toimii alle vuoden vanhassa käyttöjärjestelmäversiossa. Jatkona itse valvontaohjelman tutkimiselle oli salauksen, hälytyksien ja erityyppisillä tietoliikenteillä toimivien tapauksien tutkiminen.

# 2 KÄYTETYT TEKNOLOGIAT

## 2.1 Käyttöjärjestelmät

Itse työssä on käytetty kahta eri käyttöjärjestelmää. Näihin käyttöjärjestelmiin pääsy on kuitenkin vaatinut Windows- ja Debian 8 -järjestelmien käyttöä. Näitä kahta ei kuitenkaan ole tässä dokumentissa eroteltu, sillä niiden asetukset eivät suoranaisesti vaikuta työhön.

Käyttöjärjestelmien valitsemisella ei ole niinkään väliä, mitä tulee valvontaan. Valvontajärjestelmien tulisi muokkautua niitä tarvitsevien käyttöjärjestelmien mukaan, eikä niin, että käyttöjärjestelmät muokkautuisivat helposti valvottaviksi.

### 2.1.1 Centos 7

Centos Linux -jakelu on vakaa, ennakoitava, hallittava ja toistettavissa oleva alusta, joka on johdettu Red Hat Enterprise Linux (RHEL) -lähteistä (CentOS 2020). Centos on valittu tähän työhön esimerkiksi sen yksinkertaisuuden ja toimivuuden vuoksi.

Tämän projektin valvontaosuudessa käytettiin kahta Centos-käyttöjärjestelmällä varustettua virtuaalikonetta. Kummassakin koneessa oli minimaaliset versiot eli kummassakaan ei ole ollut käytössä graafista käyttöliittymää. Koneita hallinnoitiin komentoriviltä Linuxin komennoilla.

Tämän lisäksi virtuaalikoneympäristö on rakennettu Centos 7 -järjestelmään. Alusta ei ole ollut virtuaalikone vaan HP:n palvelin. Alustana toimi itse palvelin, sillä se tarjoaa huomattavasti enemmän tilaa kuin yksittäinen virtuaalikone. Kokonaisuus toimi projektin hypervisorina, ja sen tehtävänä oli pitää yllä ja varastoida kaikki virtuaalikoneet.

### 2.1.2 Debian 10

Debian 10 on Debian-käyttöjärjestelmä, joka tulee 59 tuhannen paketin kanssa ja perustuu Linux-ytimeen (Debian 2020). Debian 10 on uusin versio käyttöjärjestelmästä ja on julkaistu vuonna 2019. Koska käyttöjärjestelmä on niin uusi, ei kaikkiin sen käyttämiin elementteihin löydy suoraan ratkaisua etsimällä.

Debian 10 otettiin käyttöön juuri sen uutuuden takia. Tarkoituksena oli testata Prometheuksen toimivuutta uudemmissakin käyttöjärjestelmissä, sillä tästä ei löydy juurikaan tietoa.

## 2.2 KVM – Kernel-based Virtual Machine

Kernel-based Virtual Machine eli KVM on virtualisointiratkaisu. Se on rakennettu Linux-ytimeen ja sisältää virtualisointiin vaativat työkalut. Käyttämällä KVM:ää pystytään pitämään käynnissä useita virtuaalikoneita, joissa toimii Linux- tai Windows-image.

### 2.2.1 Ympäristö

KVM:n avulla voidaan asentaa virtuaalikoneita haluttuun alustaan. Tässä projektissa virtuaalikoneet oli asennettu Centos 7 -koneelle, joka toimi hypervisorina. Virtuaalikoneet tallentuvat alustalle imageina, joita voidaan tarkastella erinäisillä komennoilla. Kuvassa 1 on tarkastettu kaikki käynnissä olevat koneet.

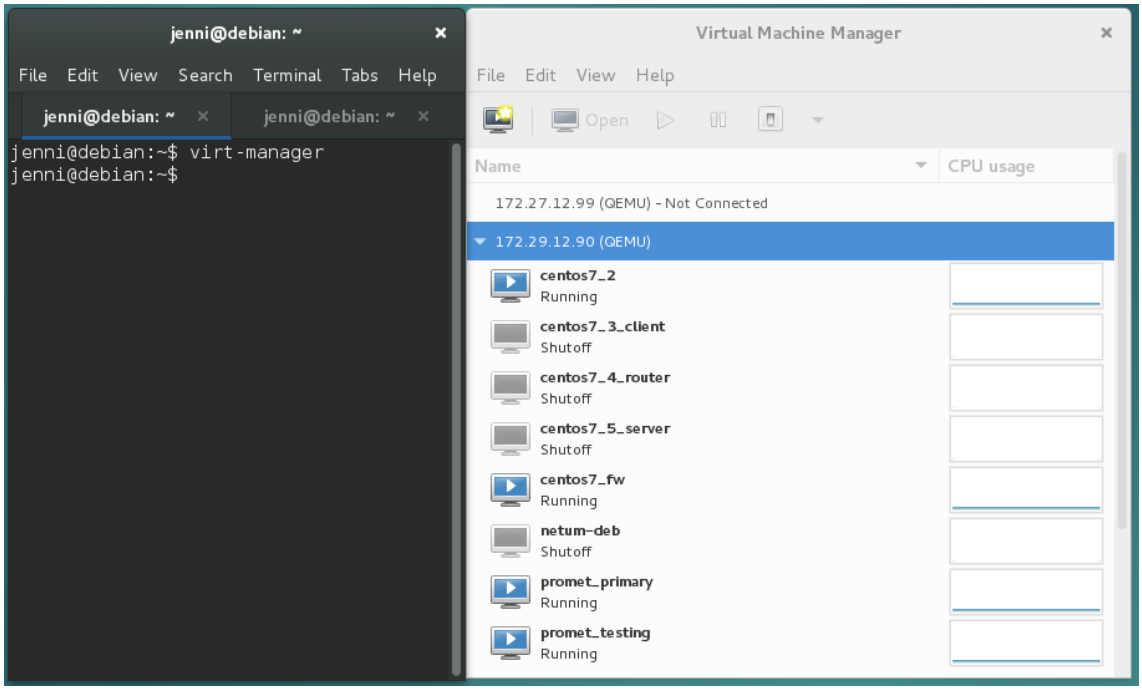
```
[admin@localhost ~]$ sudo virsh list
[sudo] password for admin:
 Id    Name                           State
----------------------------------------------------
 1     centos7_2                      running
 2     centos7_fw                     running
 4     promet_primary                 running
```

KUVA 1. Käynnissä olevat virtuaalikoneet KVM-ympäristössä

Hypervisorille voidaan helposti asentaa uusi koneita eli imageja. Näitä voidaan luoda ja hallita komentoriviltä.

### 2.2.2 Virtuaalikoneiden hallinnointi / Virtual Machine Manager (VMM)

Projektissa käytettiin Virtual Machine Manager –nimistä (VMM) ohjelmaa. Ohjelman avulla virtuaalikoneiden hallinta on helppoa. VMM:n avulla voitiin muokata esimerkiksi verkkokortteja sekä avata graafinen käyttöliittymä. Kuvassa 2 on avattu VMM ja sillä yhdistetty hypervisoriin. Tätä kautta on päästy tarkastelemaan ja käyttämään kaikkia virtuaalikoneita, mitkä ovat olleet asennettuina.



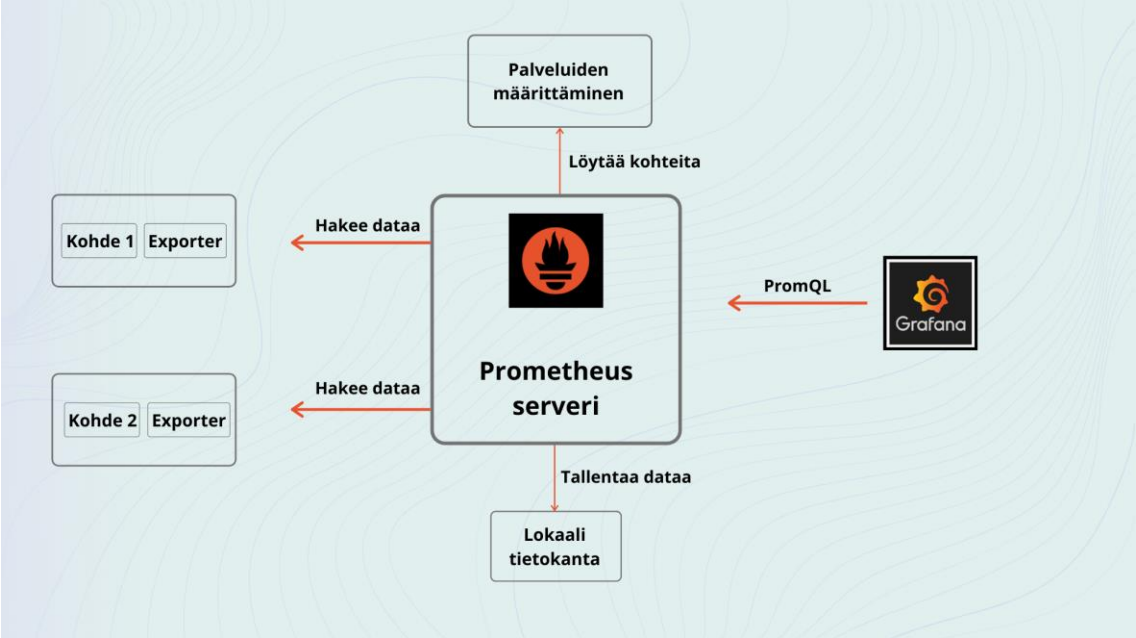

KUVA 2. Virtual Machine Managerin käyttö

## 2.3 Prometheus-ohjelma

Prometheus on Go-kielellä kirjoitettu avoimen lähdekoodin ohjelma, joka koostuu erilaisista paketeista. Näitä paketteja voidaan ladata valmiina tai voidaan luoda itse omaan tarpeeseen. Koska mitään ei ole oletuksena asennettu, on ohjelma hyvin kevyt ja helposti omaan käyttöön muokattavissa. Sen alkuperä johtaa tilanteeseen, jossa haluttiin mahdollisimman helposti muokattava valvontaohjelma. Jotta erinäisten järjestelmien yhdistely ei olisi ollut tässä tilanteessa tarpeellista,

kehitettiin Prometheus joustavaksi järjestelmäksi. Hyvään saatavuuteen päästään, kun laitetaan kaksi eri Prometheus-serveriä valvomaan samaa kohdetta ja toimittamaan tämä data hälytysjärjestelmään.

Kuva 3 esittää Prometheuksen yksinkertaisen arkkitehtuurin. Itse Prometheusta käytetään jollain käyttäjän valitsemalla alustalla. Lokaali tietokanta vaatii tilaa, ja palveluiden määrittämiseen kuuluu kohteiden määrittäminen, jotta Prometheus osaa hakea dataa oikeasta paikasta. Näillä kohteilla täytyy olla Prometheukselle tehty välittäjä asennettuna, jotta jokin keräisi kohteesta dataa ja kääntäisi sen Prometheuksen ymmärrettäväksi. Visualisointi vaatii oman kyselykielensä, jotta se osaa pyytää oikeanlaista dataa serveriltä.




KUVA 3. Prometheuksen arkkitehtuuri

Prometheus toimii pull-menetelmällä eli mikään ulkopuolinen ei voi syöttää sille tietoa, vaan se itse hakee ulkopuolelta dataa sen omista asennetuista lisäosista. Se myös tallentaa datan lokaalisti eikä esimerkiksi pilveen. Idea lokaalissa menettelyssä on ollut säästyä turhilta ongelmilta, sillä välissä käytetyt ratkaisut voivat olla niitä, jotka kaatuvat ensin.

Dataa Prometheus-serveriltä haetaan esimerkiksi visualisointiin Prometheuksen omalla kyselykielellä nimeltä PromQL (Prometheus Query Language). Kyselyillä voidaan seurata dataa oikeassa ajassa tai tietyltä aikaväliltä.

### 2.3.1 Skriptit

Prometheusta ja sen kaikkia lisäosia ajetaan skripteillä. Skriptit on luotu nimenomaan valvontaohjelman ja datan lähettäjien käynnistämiseen sekä serverien käynnistämiseen.

Skriptien sisältö on laadittu vastaamaan koko valvonnan ohjaamisesta. Skriptit ovat esimerkiksi Python-kielellä kirjoitettuja tiedostoja, jotka ajamalla tapahtuu sen tiedoston sisällä määriteltyjä toimintoja. Näitä tiedostoja pystyy luomaan itse omaan tilanteeseen sopiviksi.

### 2.3.2 Exporterit

Exporterit toimivat Prometheuksessa lisäliitoksina, jotka auttavat lukemaan oikeanlaista dataa erilaisista kohteista. Ilman näitä Prometheus lukisi vain itseään oletuksena. Exportereita on useita valideja ja Prometheuksen itse validoimia sekä yksittäisten käyttäjien tekemiä liitoksia. Jälkimmäisien käytössä tulee taas harjoittaa varovaisuutta, eikä kaikkia tule huolimattomasti ladata ja ottaa käyttöön.

Exporterien perustoiminta on lukea sille suunnattua ja konfiguroitua ominaista dataa. Tämän jälkeen sen tehtävänä on muokata se Prometheukselle sopivaksi mittaridataksi ja antaa tämä data Prometheukselle, kun valvontaohjelma sitä pyytää. Exporterit eivät itse tallenna dataa mihinkään, vaan se kuuluu valvontapalvelimen tehtäviin.

## 2.4 Grafana

Grafana antaa mahdollisuuden kysellä, visualisoida, hälyttää ja ymmärtää mittaridataa riippumatta siitä, missä ne sijaitsevat (Grafana. 2020). Grafana on vain yksi monista visualisointimahdollisuuksista. Grafanalta löytyy oma liittimensä

Prometheukseen, joten sen käytön pitäisi olla suhteellisen helppoa uudellekin käyttäjälle. Grafana toimii selaimessa.

Grafana laitetaan toimimaan sisällyttämällä sen tietoihin datan lähde. Datan lähdettä voidaan käyttää kojelaudoissa, joita voidaan ladata lisäosiksi tai luoda oma, vastaamaan siitä, miltä data tarkastelijalle näyttää.

Datan lähde ja kojelaudat säilyvät luotujen tunnuksien alla, mutta Grafanaa käytetään oletuksena lokaalisti, joten sen tarkastelu ulkopuolisesta koneesta vaatii palomuurien konfigurointia uudelleen. Tässäkin tapauksessa täytyy kuitenkin tietää tunnukset, jotta kojelautoja pääsee tarkastelemaan.

# 3 TEKNINEN TOTEUTUS

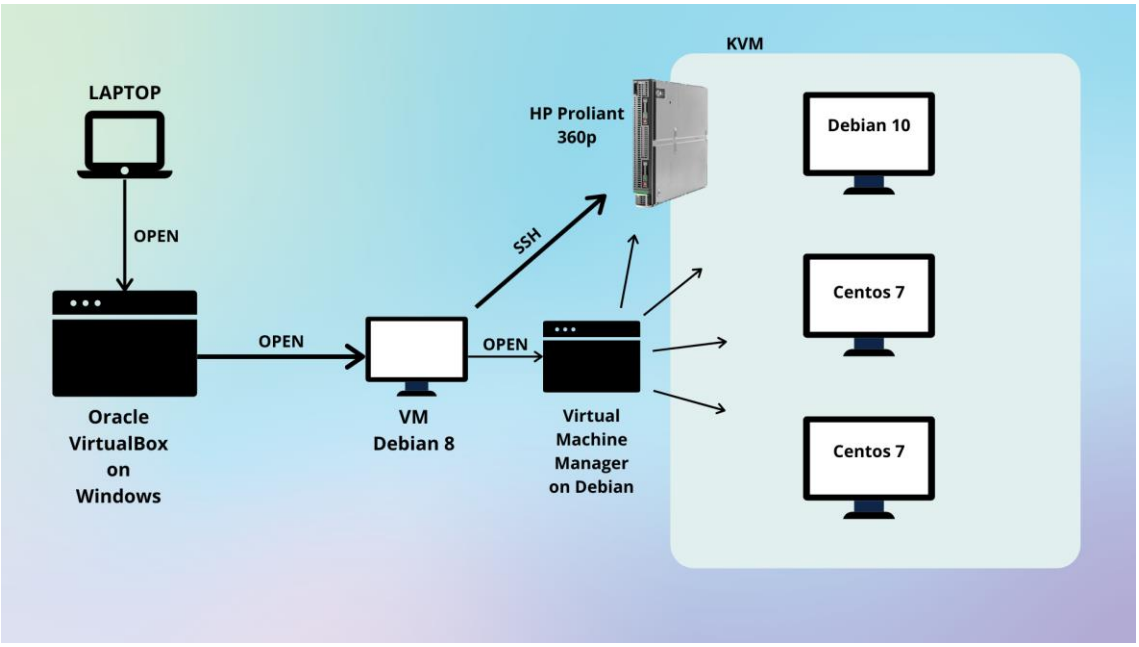
## 3.1 Arkkitehtuuri

Kuvasta 4 nähdään työssä käytettyjen alustojen arkkitehtuuri. Työtä on tehty omalta kannettavalta tietokoneelta käyttäen sovellusta nimeltä VirtualBox, jonka avulla voidaan avata virtuaalikoneita. Tässä työssä alustana on käytetty Debian 8 -käyttöjärjestelmää, johon on asennettu Virtual Machine Manager eli Virtuaalikoneiden hallinnoitsija. Tätä työkalua voidaan käyttää hallinnoimaan virtuaalikoneita SSH-yhteyden päässä.

Varsinainen valvontatyö on tehty HP Proliant -palvelimelle asennetussa käyttöjärjestelmässä, johon on asennettu KVM-ympäristö, mikä sisältää useita virtuaalikoneita. Näistä koneista on käytetty kolmea. Yksi toimi valvontaohjelman serverinä ja kaksi muuta toimivat pelkästään valvottavina kohteina.

Yhteys valvontaympäristössä toimiviin koneisiin voidaan ottaa joko omalle tietokoneelle asennetulta virtuaalikoneelta SSH-yhteydellä tai käyttäen virtuaalikoneiden hallinnoitsijasovellusta. Jälkimmäistä käyttämällä saadaan graafinen käyttöliittymä virtuaalikoneista auki sen sijaan, että konetta hallinnoitaisiin komentoriviltä.

Varsinaisen valvontaympäristön arkkitehtuuri on selitetty käytetyissä teknologioissa Prometheuksen oman arkkitehtuurin yhteydessä kuvassa 3. Näitä kahta arkkitehtuurikuvaa tarkastelemalla saadaan laajempi käsitys siitä, miten kokonaisuus toimii.




KUVA 4. Projektin työskentelyssä käytetyn ympäristön arkkitehtuuri

## 3.2 Prometheus-serveri

Prometheus asennettiin Debian 10 -virtuaalikoneelle. Koneen asetuksiin ei tarvitse välttämättä tehdä muutoksia. Tässä tapauksessa kuitenkin asennettiin wget-komento Prometheus-paketin lataamista varten.

Käyttöä varten täytyy ladata paketti Prometheuksen omilta sivuilta, kuten kuvassa 5 on valittu Linuxille sopiva paketti. Se täytyy valita käyttöjärjestelmän perusteella. Paketti on tar-tiedostomuodossa.

prometheus

The Prometheus monitoring system and time series database. prometheus/prometheus

2.18.1 / 2020-05-07 Release notes

| File name | OS | Arch | Size |
|---|---|---|---|
| prometheus-2.18.1.darwin-amd64.tar.gz | darwin | amd64 | 60.16 MiB |
| prometheus-2.18.1.linux-amd64.tar.gz | linux | amd64 | 60.44 MiB |
| prometheus-2.18.1.windows-amd64.tar.gz | windows | amd64 | 59.98 MiB |

KUVA 5. Prometheus-paketin latauslinkit

### 3.2.1 Prometheuksen asennus ja konfigurointi

Paketin latauksen jälkeen ohjelma täytyy asentaa koneelle. Ohjelma asennettiin käyttäen tar-tiedostomuodoille suunnattua komentoa, joka purkaa ja asentaa paketin. Kuvassa 6 on navigoitu kansioon, mikä on tehty Prometheusta varten. Kansiossa ajetaan komento, jolla avataan ja asennetaan tar-paketti. Tämän jälkeen voidaan navigoida kansioon, jonka sisältä löytyy kaikki tarvittava Prometheuksen ajamiseen.

```
promeetta@promet:/etc/prometheus$ tar xvfz prometheus-2.18.1.linux-amd64.tar.gz
promeetta@promet:/etc/prometheus$ cd prometheus-2.18.1.linux-amd64/
promeetta@promet:/etc/prometheus/prometheus-2.18.1.linux-amd64$ ls
console_libraries  consoles  LICENSE  NOTICE  prometheus  prometheus.yml  promtool  tsdb
```

KUVA 6. Prometheuksen asennus ja ohjelmakansion sisältö

Kansion tärkeimpiä tiedostoja ovat prometheus.yml ja prometheus. Ensimmäistä muokkaamalla Prometheus saadaan valvomaan haluttuja kohteita ja jälkimmäinen on skripti. Kun skripti ajetaan, serveri käynnistyy valvomaan.

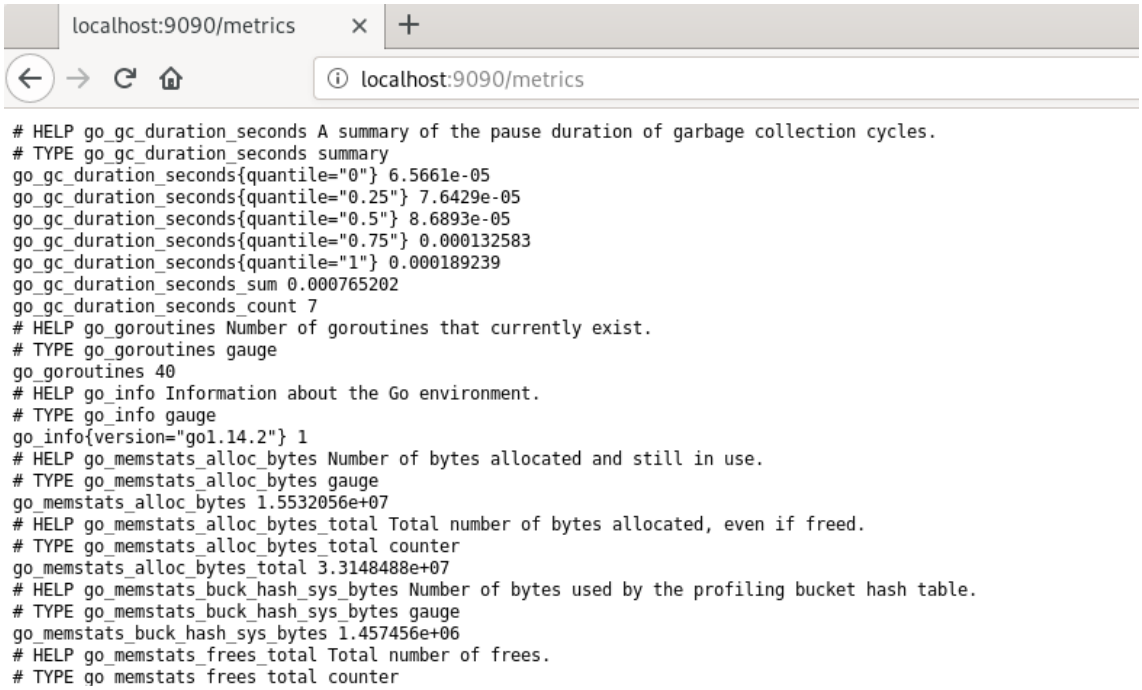
### 3.2.2 Prometheus valvomassa itseään

Prometheuksen toiminta testattiin laittamalla se valvomaan itseään. Konfigurointitiedoston sisältö on oletukseltaan lähes kuvan 7 näköinen. Tässä tapauksessa tiedostosta kuitenkin poistettiin alkuun turhat sisällöt kuten sääntöjen tarkastaja sekä hälytyksien hallinnoija. Globaalit tiedot määrittävät esimerkiksi sen, kuinka usein dataa haetaan asetetuista kohteista. Scrape configs, eli kaavittavan datan konfiguroinnit, määrittävät töiden nimet ja sijainnit, mistä dataa niin sanotusti kaavitaan.

```
global:
  scrape_interval:     15s # By default, scrape targets every 15 seconds.

  # Attach these labels to any time series or alerts when communicating with
  # external systems (federation, remote storage, Alertmanager).
  external_labels:
    monitor: 'codelab-monitor'

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: 'prometheus'

    # Override the global default and scrape targets from this job every 5 seconds.
    scrape_interval: 5s

    static_configs:
      - targets: ['localhost:9090']
```

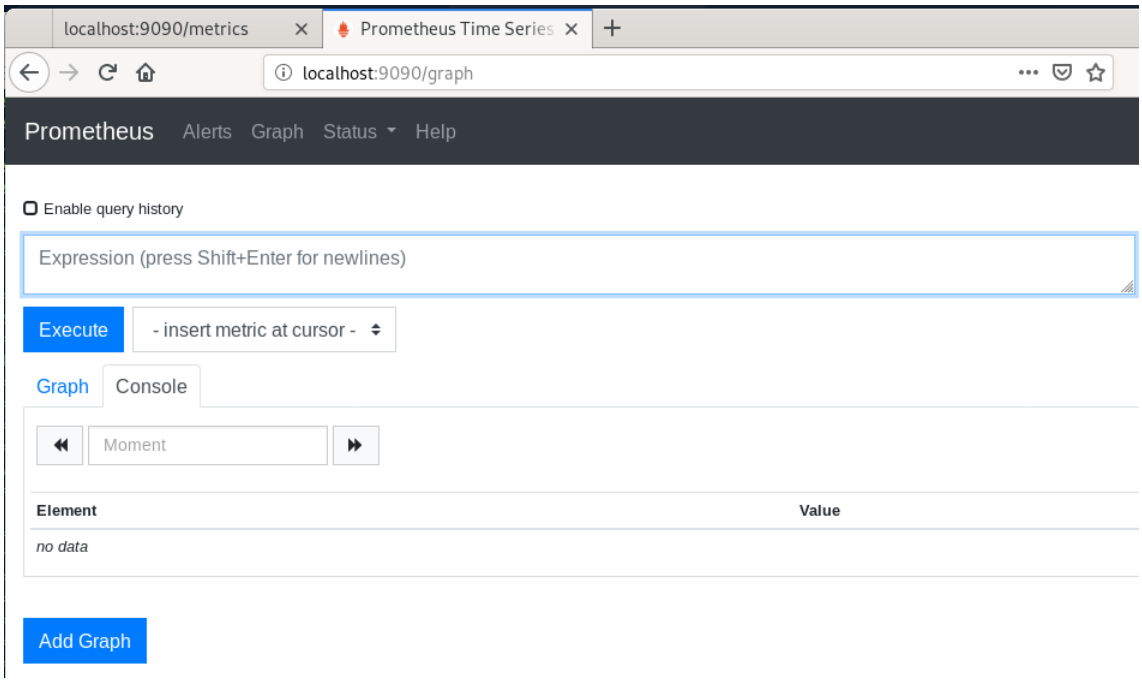
KUVA 7. Prometheus.yml-tiedoston oletussisältö

Oletussisällön avulla Prometheus valvoo itseään. Prometheus toimii oletuksena portissa 9090. Käynnistys vaatii skriptin ajamisen, jolloin terminaalin pitäisi ilmoittaa, että serveri on valmis ottamaan vastaan verkkokyselyitä. Tämän datan mittareita voidaan tarkastella syöttämällä selaimen hakuriville localhost:9090/metrics. Tämä sivusto antaa kuvan 8 mukaista tietoa.




KUVA 8. Prometheuksen datan mittarit

Luonnollisesti sivu antaa konfiguroinnin mukaisesti 15 sekunnin välein uutta dataa, kun Prometheus kaapii uudestaan. Mittareiden tarkastelu ei kuitenkaan ole aina se, mitä halutaan, ja tätä varten voidaan kirjoittaa selaimen hakukenttään localhost:9090/. Tällöin kuvan 9 mukainen Prometheuksen oma sisäänrakennettu graafinen käyttöliittymä aukeaa. Tämä sisäänrakennettu liittymä on hyvin alkukantainen ja ei ole hyödyllinen laajempaa valvontaa tarvitseville kohteille, mutta toimii hyvin nopeaa tarkastelua vaativille kohteille.




KUVA 9. Prometheuksen graafinen käyttöliittymä

Kuvasta 9 nähdään, että voidaan vaihdella konsolin ja kuvion välillä. Käyttöliittymään voidaan syöttää Prometheuksen oman kyselykielen mukaisesti komentoja, mikäli tiedot halutaan rajata tietyn datan mukaan, kuten muistin käytön mukaan. Kyselykielen käyttöä varten pitää kääntyä ohjelman omien oppaiden suuntaan. Lyhyesti kyselykieli toimii siten, että sille annetaan komento ja sulkujen sisälle syötetään jokin rivi mittareiden sisältä. Kyselylle voidaan antaa myös aikaväli, miltä dataa tarkastellaan.

Itsensä valvomisen lisäksi on tärkeää saada muutakin tietoa serveriltä, kuin vaan ohjelman omasta suorituksesta. Samalle koneelle asennettiin myös esimerkiksi Node_exporter, jotta voitiin valvoa myös valvontaohjelmaa ylläpitävän koneen

suoritusta. Tämän datan kuljettajan asennuksesta kerrotaan enemmän kappaleessa 3.3.

### 3.2.3 Ulkopuolisten datanlähteiden konfigurointi

Ulkopuolisilla datan lähteillä tarkoitetaan Prometheus-serverin ulkopuolelta tulevaa dataa, kuten toisesta koneesta. Jotta valvontaohjelma osaisi pyytää tietoa toiselta koneelta, täytyy se konfiguroida itse valvontaohjelmaan. Kaikki konfigurointi tapahtuu samaan tiedostoon, mihin myös Prometheuksen oma valvonta konfiguroitiin.

Konfigurointitiedostoon lisätään töitä sitä mukaa kun uusia kohteita asennetaan. Kuvasta 10 nähdään, että työlle on annettu nimi, ja sille on kerrottu, missä kohde sijaitsee. Tässä esimerkissä kohde sijaitsee samalla koneella, ja siksi kohteelle on annettu sijainniksi localhost. Mikäli kohde sijaitsee toisella koneella, laitetaan tilalle sen koneen IP. Portti 9100 on Node-exporterin oletusportti.

```
- job_name: 'node_exporter'
  static_configs:
  - targets: ['localhost:9100']
```

KUVA 10. Prometheuksen konfiguroiminen pyytämään dataa ulkoisesta lähteestä

## 3.3 Node Exporter

Node Exporter eli datan kuljettaja toimii ikään kuin Prometheuksen agenttina. Sen tehtävänä on kerätä dataa ja lähettää se eteenpäin Prometheukselle, joka pystyy käsittelemään annettua dataa. Koska data tulee mittareina, ei siitä synny suoranaista lokia.

Node Exporterin lataukseen sekä asennukseen käytetään samaa metodia kuin itse Prometheuksen. Tar-tiedosto ladataan luotettavasta lähteestä koneelle. Tämä kyseinen paketti tai lisäosa on tarkoitettu valvomaan Linux-koneita. Mikäli

halutaan valvoa eri järjestelmiä tai sovelluksia, täytyy näille valita eri paketti tai luoda täysin uusi omaan tilanteeseen sopiva datan käsittelijä.

Node Exporter tulee sisältäen kaikki perustarpeet, mitä Linux-koneesta yleensä halutaan tietää. Tässä tapauksessa muistin käyttö ja koneen suoritus olivat tarkastelun alla.

### 3.3.1 Datan kuljettajan asennus ja käyttö

Datan kuljettaja asennettiin kahteen Centos 7 -koneeseen. Tar-tiedosto avataan samalla tar-komennolla, kuten Prometheuksen oma asennuspaketti. Kuvassa 11 ovat komennot sekä tarvittavien kansioiden sisällöt. Itse Exporterin kansiosta löytyy infotiedostojen lisäksi ainoastaan skripti, millä datan kuljettaja käynnistetään. Tämä tapahtuu kuvan viimeisellä komennolla.

```
promeetta@promet:/etc/prometheus$ sudo tar xvfz node_exporter-0.18.1.linux-amd64.tar.gz
promeetta@promet:/etc/prometheus$ ls
node_exporter-0.18.1.linux-amd64         prometheus-2.18.1.linux-amd64
node_exporter-0.18.1.linux-amd64.tar.gz
promeetta@promet:/etc/prometheus$ cd node_exporter-0.18.1.linux-amd64/
promeetta@promet:/etc/prometheus/node_exporter-0.18.1.linux-amd64$ ls
LICENSE  node_exporter  NOTICE
promeetta@promet:/etc/prometheus/node_exporter-0.18.1.linux-amd64$ ./node_exporter
```

KUVA 11. Node_exporterin asennus ja käynnistys

Dataa voidaan tarkastella selaimessa. Tämän tapahtuu syöttämällä selaimen komentoriville localhost:9100/metrics, jolloin nähdään datan mittarit, kuten muistin määrä.

### 3.3.2 Palomuurit

Ainakin Centos 7 -järjestelmässä on oletuksena portti 9100 suljettu. Portti täytyy tämän vuoksi konfiguroida palomuurisääntöihin, jotta data saataisiin liikkumaan Prometheus-serverille.

```
sudo yum install nmap
sudo firewall-cmd --add-port=9100/tcp --permanent
systemctl restart firewalld
```

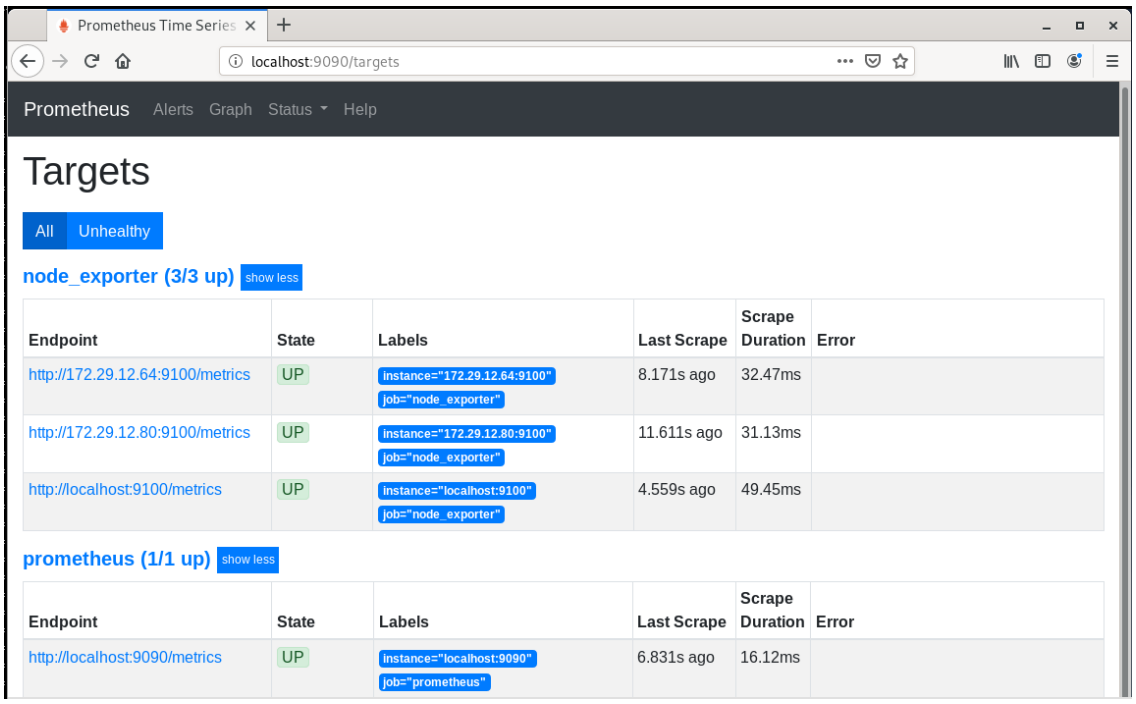
KUVA 12. Palomuurin konfigurointi

Kuvassa 12 ovat työssä tehdyt muutokset palomuuriin, jotta portti 9100 saataisiin auki. Ensin on asennettu työkalut tätä varten, sitten lisätty portti sääntöihin ja tämän jälkeen palomuuri on käynnistetty uudestaan. Sääntöjä muutettaessa saatetaan törmätä ongelmaan, missä palomuuri on naamioitu. Tällöin komento systemctl unmask --now firewalld, paljastaa palomuurin ja sääntöjen muokkaamista voidaan jatkaa.

Palomuurien sääntöjen muokkauksessa tulee kuitenkin yleisesti huomioida varovaisuus. Mikäli sääntöjä muutetaan mielivaltaisesti, voidaan mahdollisesti avata ovet luvattomille vierailijoille.

### 3.3.3 Yhteys Prometheukseen

Kun datan kuljettaja on käynnistetty, lisätään kuvan 10 mukaisesti Prometheuksen omaan konfigurointitiedostoon Centos 7 -koneen IP-osoite. Valvontaohjelma käynnistetään uudelleen tai konfigurointitiedosto ladataan uudelleen. Selaimen hakukenttään syötetään taas osoite localhost:9090/, jolloin Prometheuksen graafinen käyttöliittymä aukeaa. Tämän jälkeen navigoidaan yläpalkista kohteet välilehdelle, josta nähdään, kuvan 13 mukaisesti, onko linkki ylhäällä.




KUVA 13. Prometheuksen valvomat kohteet oikean konfiguroinnin jälkeen

Kuvan 13 mukainen tilanne tapahtuu vain, mikäli Node Exporterit ovat toiminnassa kaikilla koneilla ja Prometheus serveri on käynnistetty. Mikäli linkki on alhaalla, on joko konfigurointi tai käynnistys epäonnistunut. Koneiden palomuurit olivat myös tiellä. Centos 7 -koneiden palomuureissa on oletuksena estetty portti 9100, joten käyttöjärjestelmiin tehtävät asetukset täytyy alustuksessa ottaa huomioon.

## 3.4 Visualisointi

Visualisointi on tehty Grafana-visualisointiohjelmalla. Jotta visualisointi onnistuisi, täytyy datan lähteet olla tiedossa, sekä tietää, millä tavalla data halutaan visualisoida. Tässä tapauksessa haluttiin seurata virtuaalikoneiden suoritusta.

### 3.4.1 Käytön aloitus

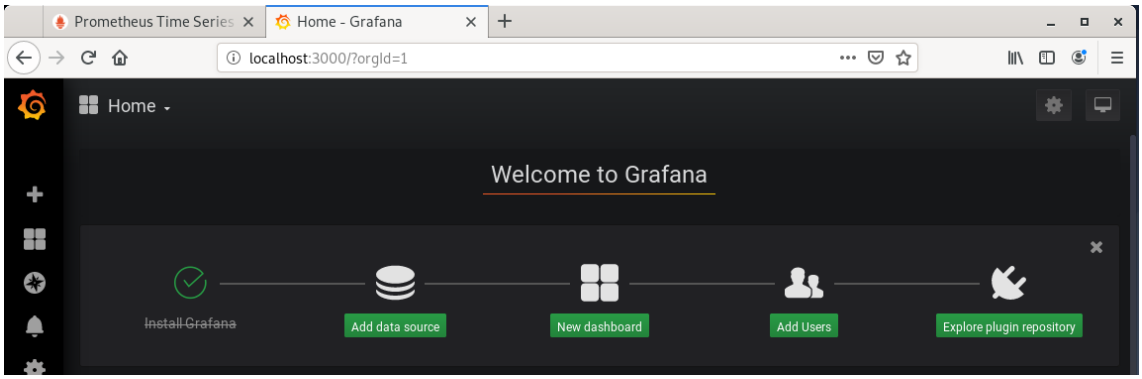
Visualisointiohjelman käyttöönotto tapahtui lataamalla ja asentamalla Grafana Prometheus-serverille. Lataaminen onnistuu Grafanan omilta verkkosivuilta. Kuvassa 14 Grafana ladataan, asennetaan ja käynnistetään. Lataaminen tapahtui

wget-komennolla, paketin avaaminen sekä asentaminen tar-komennolla ja käynnistys ajamalla skripti.

```
promeetta@promet:/etc/prometheus$ wget
https://dl.grafana.com/oss/release/grafana-7.0.0.linux-amd64.tar.gz
promeetta@promet:/etc/prometheus$ sudo tar xvfz grafana-6.7.3.linux-amd64.tar.gz
promeetta@promet:/etc/prometheus/grafana-6.7.3$ sudo ./bin/grafana-server
```

KUVA 14. Grafanan asennus ja käynnistys

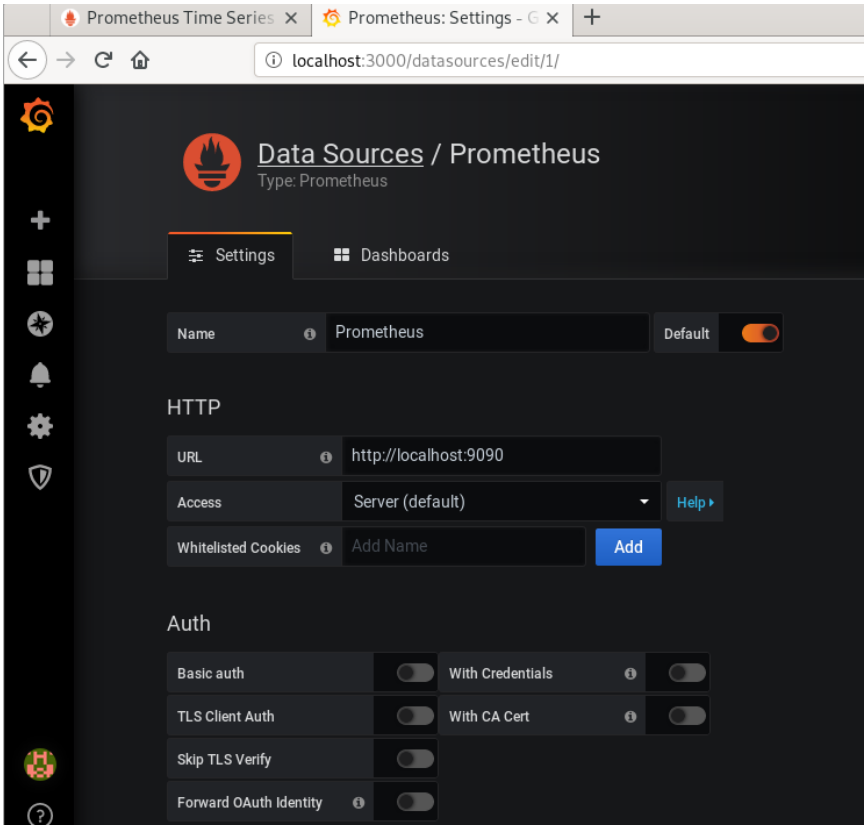
Grafana kuuntelee porttia 3000, joten serverin käynnistyksen jälkeen voidaan navigoida osoitteeseen localhost:3000/. Sivulle aukeaa ensimmäisenä kirjautumissivu, jossa luodaan ensimmäisenä tunnukset. Kun kirjautuminen on onnistunut kuvan 15 mukaisesti, täytyy Grafana konfiguroida toimimaan tarpeiden mukaisesti.




KUVA 15. Grafanan etusivu

### 3.4.2 Datan lähteet

Tietoa haluttiin visualisoida kolmesta koneesta. Kaikki kolme konetta on konfiguroitu Prometheuksen sisään, joten antamalla Prometheus-serverin tiedot saatiin yhteys kaikkiin datanlähteisiin. Konfigurointi tapahtui, kuten kuvassa 16 näkyy. Koska Prometheus oli käynnissä samalla koneella kuin Grafana, annettiin lähteen osoitteeksi localhost. Kun tiedot on syötetty, klikataan tallenna ja lopeta. Mikäli yhdistäminen datan lähteeseen onnistui, Grafana ilmoittaa asiasta heti.

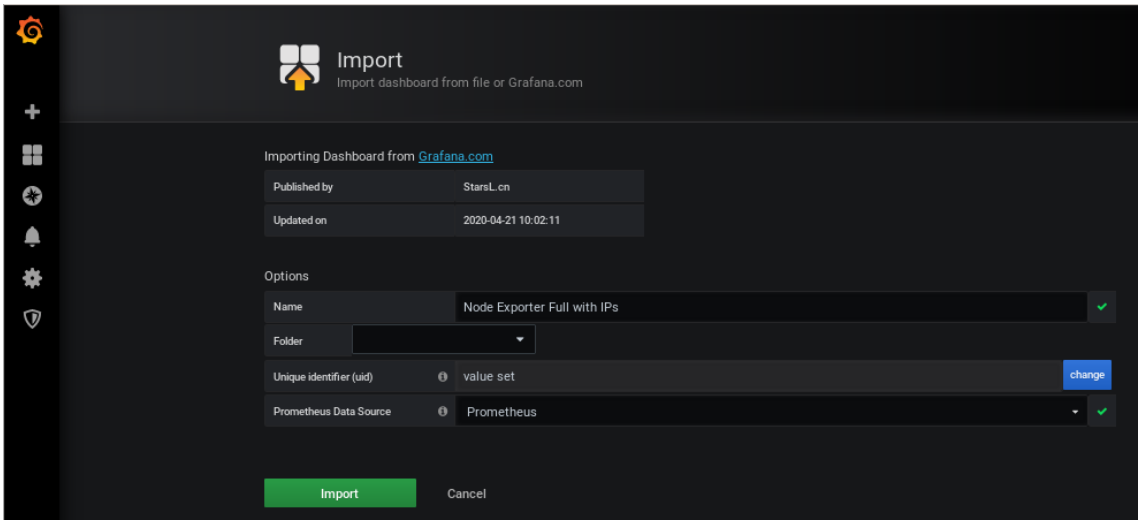



KUVA 16. Datalähteen konfigurointi Grafanaan

Datan lähteen tietäminen ei itsessään riitä Grafanalle. Grafana on Open source-ohjelma, joten käyttäjän on itse mahdollista kehittää visualisoiva kokonaisuus. Prometheukselle kuitenkin löytyy valmiita kojelautoja, joita voitiin tässä projektissa hyödyntää.

### 3.4.3 Kojelaudat

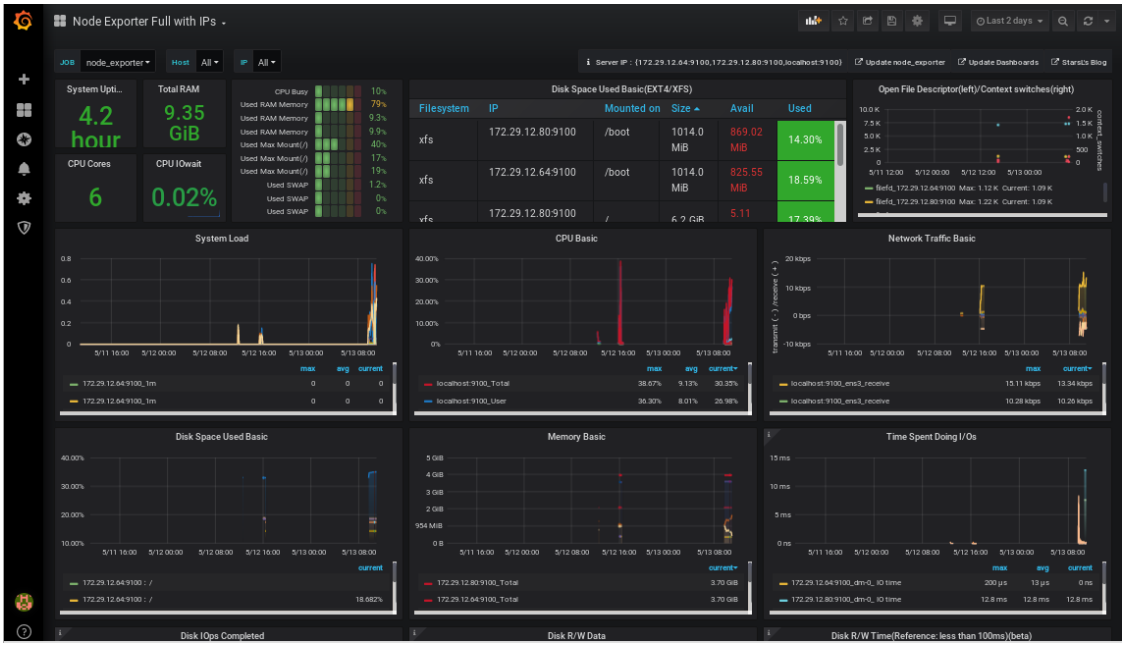
Kojelaudat toimivat visualisoivina osuuksina. Kojelautoja voidaan asentaa useampi erilainen. Tässä työssä niitä asennettiin kaksi kappaletta ajamaan eri näkökulmia. Kojelaudat asennetaan tuomalla lisäosia Grafanaan ja tämä onnistui syöttämällä sille lisäosan tunnusnumero. Tuomisen yhteydessä kojelaudalle täytyi antaa tiedoksi, mitä datanlähdettä sen tulisi käyttää, kuten kuvasta 17 näkyy.




KUVA 17. Kojelautalisäosan tuominen Grafanaan

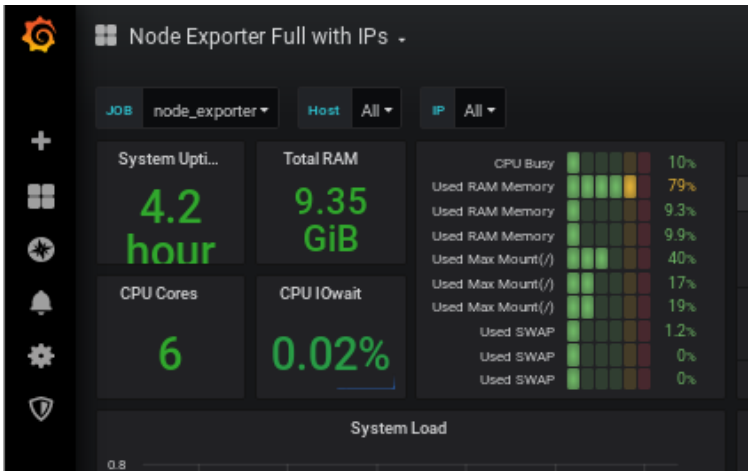
Ensimmäinen kojelauta pitää sisällään yksinkertaisen visualisoinnin. Sen sisältö sopi pienempään yhden koneen tarkasteluun graafisessa muodossa. Se kertoi mittarien arvot graafeissa. Tarvittiin kuitenkin laajempi näkymä siitä, mitä koneet pitävät sisällään.

Toiseksi kojelaudaksi valittiin laaja usean Linux-koneen valvomista varten suunniteltu kojelauta. Kuvassa 18 on näkymä kojelaudan koko sisällöstä. Tästä nähdään, kuinka data on visualisoitu sekä numeroin että graafein ja otsikoitu selkeästi.




KUVA 18. Grafanaan suunniteltu kojelauta tarkoitettu näyttämään useamman serverin suoritus

Kuvasta 19 nähdään, kuinka kojelauta käyttää kaikkea dataa, mitä Prometheus valvoo. Valikosta voitaisiin valita myös vain yksi IP-osoite, jolloin voitaisiin tarkastella yhden koneen suoritusta.




KUVA 19. Prometheuksesta seulottu työn nimi ja tarkasteltavat kohteet

Nyt Prometheusta käyttäen voidaan valvoa ja tarkastella virtuaalikoneiden suoritusta. Koneiden lisääminen on helppoa ja visualisointi ei kärsi koneiden lisäämisestä.

# 4 POHDINTA

Vaikka valvonta käyttäen Prometheus-valvontaohjelmaa on helppoa ja helposti muokattavaa, on sen käytössä otettava huomioon sen tuomat huonot puolet. Kaikki se vapaus, mitä Prometheus tarjoaa, luo tilanteen, jossa käyttäjän tulee osata ja ymmärtää sen toiminta hyvin, jotta käyttö olisi tehokasta ja turvallista.

Prometheuksen yksi huonoimmista puolista on sen tietoturvan taso. Mikäli käyttäjä ei ole osaava ja hakee yhden asennettavan paketin ratkaisua, ei tämä ohjelma ole sopiva. Integroimisen ja liikenteen valvomisen ymmärrys on tärkeää sen käytössä. Prometheus on ottanut tietoturvan ja salauksen ohjelmansa kehityssuunnitelmaan, mutta aikataulusta ei ole tässä vaiheessa tietoa. Toistaiseksi käyttäjien tulee järjestää nämä itse. Onneksi näihinkin on valmiiksi asennettavia lisäosia, mutta niidenkään yhteensopivuus ei ole taattu. Uusien käyttöjärjestelmien ja testaamattomien lisäosien tietoturvaa tulisi myös katsoa arvostellen. Salasanasuojausta yritettiin tähänkin työhön saada, mutta käyttöjärjestelmän uutuuden vuoksi ei löydetty ratkaisua käänteisen proxyn ongelmiin. Suojauksen järjestäminen olisi seuraava askel.

Valvonnan tarpeisiin kuuluu myös yleensä tarve saada hälytys esimerkiksi järjestelmän kaatuessa. Valvontaohjelman pitäisi pystyä kertomaan pikaisesti ylläpitäjälle, mikäli järjestelmissä on vika varsinkin tilanteessa, jossa ylläpitäjällä on useita järjestelmiä valvottavanaan. Nykypäivänä monitorin edessä fyysinen valvominen ei ole tarpeellista, sillä näitä hälytysmenetelmiä on saatavissa. Prometheukselle on oma lisäosa nimeltä Alertmanager, joka toimii hälyttäjänä. Ennen sen käyttöönottoa salaus olisi kuitenkin tärkeä saada toimimaan. Valvontaa ja hälytyksiä ei kannata järjestelmissä käyttää, mikäli kenellä tahansa on niihin pääsy ja siihen tilanteeseen ei tässäkään työssä haluttu joutua.

Tärkeimpiä valvottavia kohteita ovat toki tietoliikenne ja järjestelmien kommunikointi. Liikenteen valvomisessa on se hyöty, että nähdään mikäli epäilyttävää liikettä ilmenee. Tämä voisi olla merkki esimerkiksi luvattomasta vieraasta palvelimilla.

**LÄHTEET**

Centos. 2020. CentOS Linux. Luettu 29.4.2020. https://www.centos.org/about/

Debian. 2020. About Debian. Luettu 29.4.2020. https://www.debian.org/intro/about

Grafana. 2020. The analytics platform for all your metrics. Luettu 12.3.2020. https://grafana.com/grafana/

NTC Hosting. 2020. Script. Luettu 23.4.2020. https://www.ntchosting.com/encyclopedia/scripting-and-programming/script/

Prometheus. 2019. Download. Luettu 15.12.2019. https://prometheus.io/download/

Prometheus. 2019. Exporters and Integrations. Luettu 15.12.2019. https://prometheus.io/docs/instrumenting/exporters/

Red Hat. 2020. Kernel Virtual Machine. Luettu 29.4.2020. https://www.linux-kvm.org/page/Main_Page

Volz, J. 2017. Monitoring, the Prometheus Way. Youtube-video. Katsottu 8.5.2020. https://www.youtube.com/watch?v=PDxcEzu62jk

Xie, H. 7.11.2018. How to Fix “No Route to Host” in Prometheus node_exporter. Luettu 3.4.2020. https://www.henryxieblogs.com/2018/11/how-to-fix-no-route-to-host-in.html